FCC-hh analysis chain

Clement Helsens, CERN
Monte Carlo Event Generation

Delphes FastSim
FCC-hh Framework

1. GridPack producer (adapted from CMS)
   - Makes MG5_aMC@NLO Grid Packs

2. LHE Producer
   - Produce LHE files on LSF/condor queues
   - About a 2 billion events produced

3. FCCSW
   - Runs Pythia8 parton shower+hadronisation and Delphes with FCC detector

4. Analysis pre-selection
   - Python framework produces flat ROOT trees

5. Analysis final selection
   - Python framework for optimising analysis cut flows and producing

6. Limit setting
   - Atlas inspired tool for limits and significance
   - Number of analyses using the framework

Fully integrated analysis chain
Links between various steps

Get the cross sections

---

Create a database of LHE events
Can produce ~10M events/hour

Create a database of FCC events
Can produce ~1M events/hour

Use the events in the database to produce analyses templates

Use the events produced in heppy to create stacked plots
1. Generation


- Start from Madgrap gridpacks but no problem to use other gridpacks
  - Create LHE files and store them on eos
  - List of available samples and statistics for 100TeV (also started 27TeV)

- Can also register to the database your own LHE files, they just need to comply with simple formatting rules
2. Simulation (FCCSW Delphes)


- From the LHE files, create FCC EDM files for a given Delphes parametrisation
- Also possible to directly simulate events with Pythia8
  - Useful for bunch of signals

- List of available samples and statistics is available for FCC 100TeV:

- Also started for HE-LHC 27TeV
3. Flat trees (heppy)

- [github.com/HEP-FCC/heppy](https://github.com/HEP-FCC/heppy)

- Within heppy create an analysis directory that contains always the same files:
  - Analysis.py
    - defines the list of modules to be run and the list of samples over which to run the inputs file lists and cross sections etc... is centrally defined and supported (from step 2.)

```python
from heppySampleList_fcc_v01 import *
from heppySampleList_fcc_v02 import *
#from heppySampleList_cms import *

selectedComponents = [
    pp_Zprime_5TeV_1l,  
    pp_Zprime_10TeV_1l,  
    pp_Zprime_15TeV_1l,  
    pp_Zprime_20TeV_1l,  
    pp_Zprime_25TeV_1l,  
    pp_Zprime_30TeV_1l,  
    pp_Zprime_35TeV_1l,  
    pp_Zprime_40TeV_1l,  
    pp_Zprime_45TeV_1l,  
    pp_Zprime_50TeV_1l,  
    pp_ee_lo,            
    pp_mumu_lo,          
]```
3. Flat trees (heppy)

- [https://github.com/HEP-FCC/heppy](https://github.com/HEP-FCC/heppy)

- Within heppy create an analysis directory that contains always the same files:
  - Selection.py -> to define the list of pre.selections (optional)

```python
def process(self, event):
    self.counters['cut_flow'].inc('All events')

    # select events with at least 2 leptons
    if len(event.selected_electrons)<2 and len(event.selected_muons)<2:
        return False
    self.counters['cut_flow'].inc('At least 2 same flavor leptons')

    return True
```
3. Flat trees (heppy)

- [https://github.com/HEP-FCC/heppy](https://github.com/HEP-FCC/heppy)

- Within heppy create an analysis directory that contains always the same files:
  - TreeProducer.py -> To define the variables to be stored in the output file
- Analysis flow are fully reproducible, need the outputs to be stored
  - /eos/experiments/fcc/hh/analyses/
  - /eos/experiments/fcc/helhc/analyses/

- A sub-dir is an analysis
4. Flat Tree analyser

- [https://github.com/FCC-hh-Framework/FlatTreeAnalyzer](https://github.com/FCC-hh-Framework/FlatTreeAnalyzer)
- From the files produced in 3.
  - plots and histograms for final analysis for different selections
- Selection based on variables available in output tree
- Templates on [github](https://github) so that we can fully reproduce the results

```python
signal_groups = collections.OrderedDict()
signal_groups['m_{Z} = 15 TeV'] = ['pp_Zprime_15TeV_ll']

background_groups = collections.OrderedDict()
background_groups['Drell-Yann'] = ['pp_mumu_lo']

# base pre-selections
selbase = 'lepl_pt > 200. && lep2_pt > 200. && abs(lep1_eta) < 4 && abs(lep2_eta) < 4 && zprime_ele_m>2000'

selections = collections.OrderedDict()
selections['m_{Z} = 15 TeV'] = []
selections['m_{Z} = 15 TeV'].append(selbase)
```
5. Limit, Significance

- [GitHub Link](https://github.com/FCC-hh-framework/FCCFitter)
- ROOSTAT profile likelihood ratio
  - Asymptotic limits
  - Could add any kind of uncertainties (shape or normalizations)
  - Combine channels and correlate nuisance parameters
- Full plotting available to produce:
  - Post-fit plots
  - Ranking plots
  - Correlation matrix
  - Pruning of systematics
- Other higher level macros to:
  - Produce confidence level exclusion plots versus mass
  - Scan the luminosity to get discovery potential
**Simple example $Z' \rightarrow ll$**

### ee

<table>
<thead>
<tr>
<th>process</th>
<th>yield (30.0 ab$^{-1}$)</th>
<th>stat. error</th>
<th>raw</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_Z = 40$ TeV</td>
<td>5.1</td>
<td>0.0</td>
<td>77417</td>
</tr>
<tr>
<td>Drell-Yann</td>
<td>128.2</td>
<td>0.1</td>
<td>1811513</td>
</tr>
</tbody>
</table>

### $\mu\mu$

<table>
<thead>
<tr>
<th>process</th>
<th>yield (30.0 ab$^{-1}$)</th>
<th>stat. error</th>
<th>raw</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_Z = 40$ TeV</td>
<td>6.5</td>
<td>0.0</td>
<td>98187</td>
</tr>
<tr>
<td>Drell-Yann</td>
<td>141.4</td>
<td>0.1</td>
<td>2449714</td>
</tr>
</tbody>
</table>

---

**FCC-hh Simulation (Delphes)**

- $\sqrt{s} = 100$ TeV
- $L = 30$ ab$^{-1}$

- $m_Z = 40$ TeV
- Drell-Yann

![Graph](image-url)
Simple example $Z'\rightarrow ll$

50% uncertainty on Drell-Yann Normalisation
Well constrained by profile likelihood

Reach up to 40TeV this very simple case!
Conclusions

- We provided a **simple, highly modular framework** for performing fast detector simulation

- **Integrated** in MG5 suite and in the FCCSW framework and can be used for FCC and HE-LHC studies

- Can be used and configured for:
  - quick **phenomenological** studies
  - as an **alternative for full-simulation** if accurately tuned

- Reproducibility exists, but could be improved

- Already ~20 analyses using this workflow

- Notebooks are being made available:
  - [http://swan.web.cern.ch/content/future-circular-collider-fcc](http://swan.web.cern.ch/content/future-circular-collider-fcc)